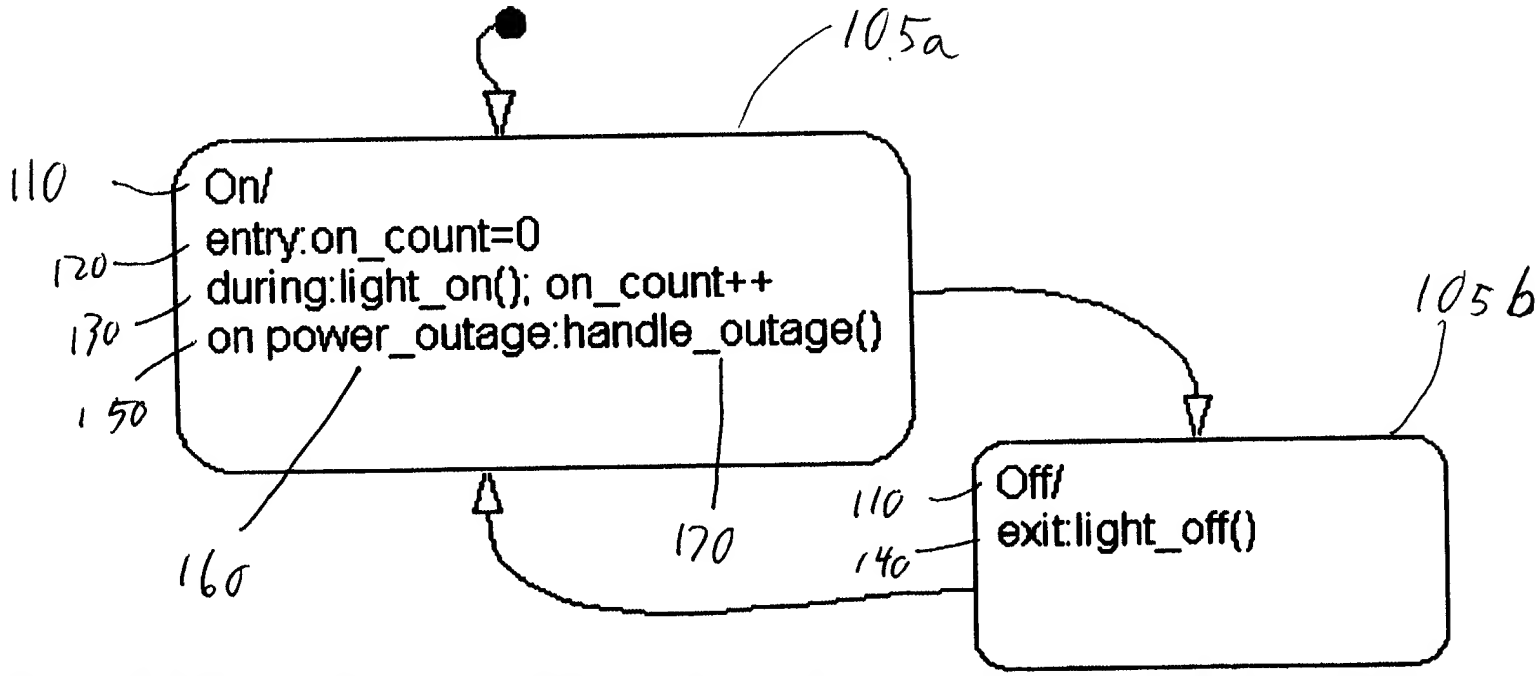
Figure 1


100
105a
110
On/
entry:on_count=0
120
during:light_on(); on_count++
130
on power_outage:handle_outage()
150
160
170
105b
110
Off/
exit:light_off()
140

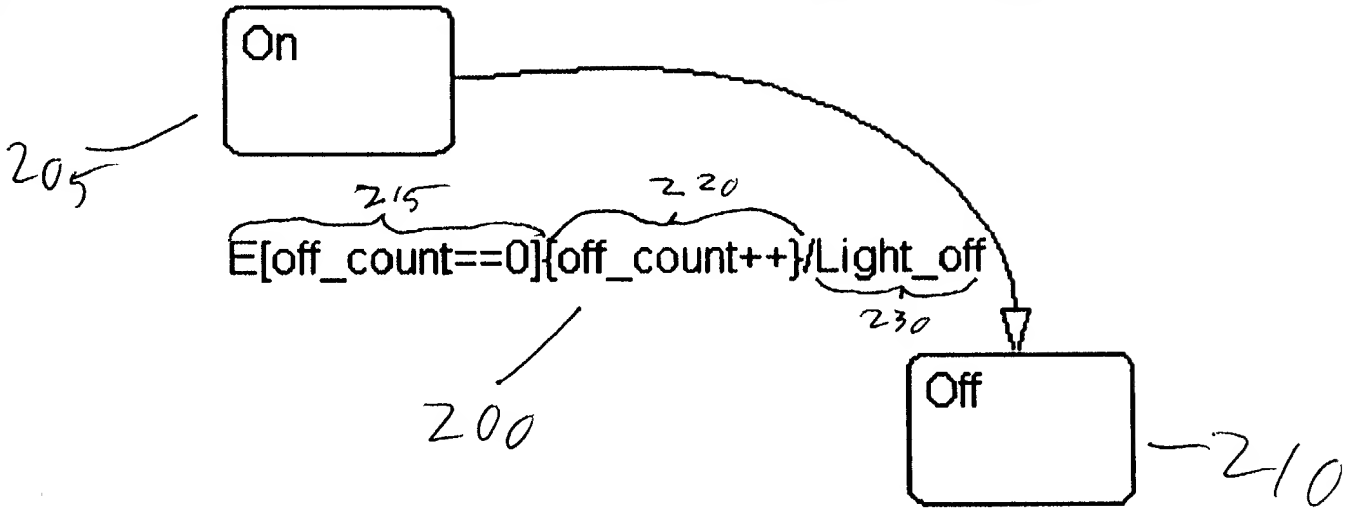


Figure 2

Figure 3

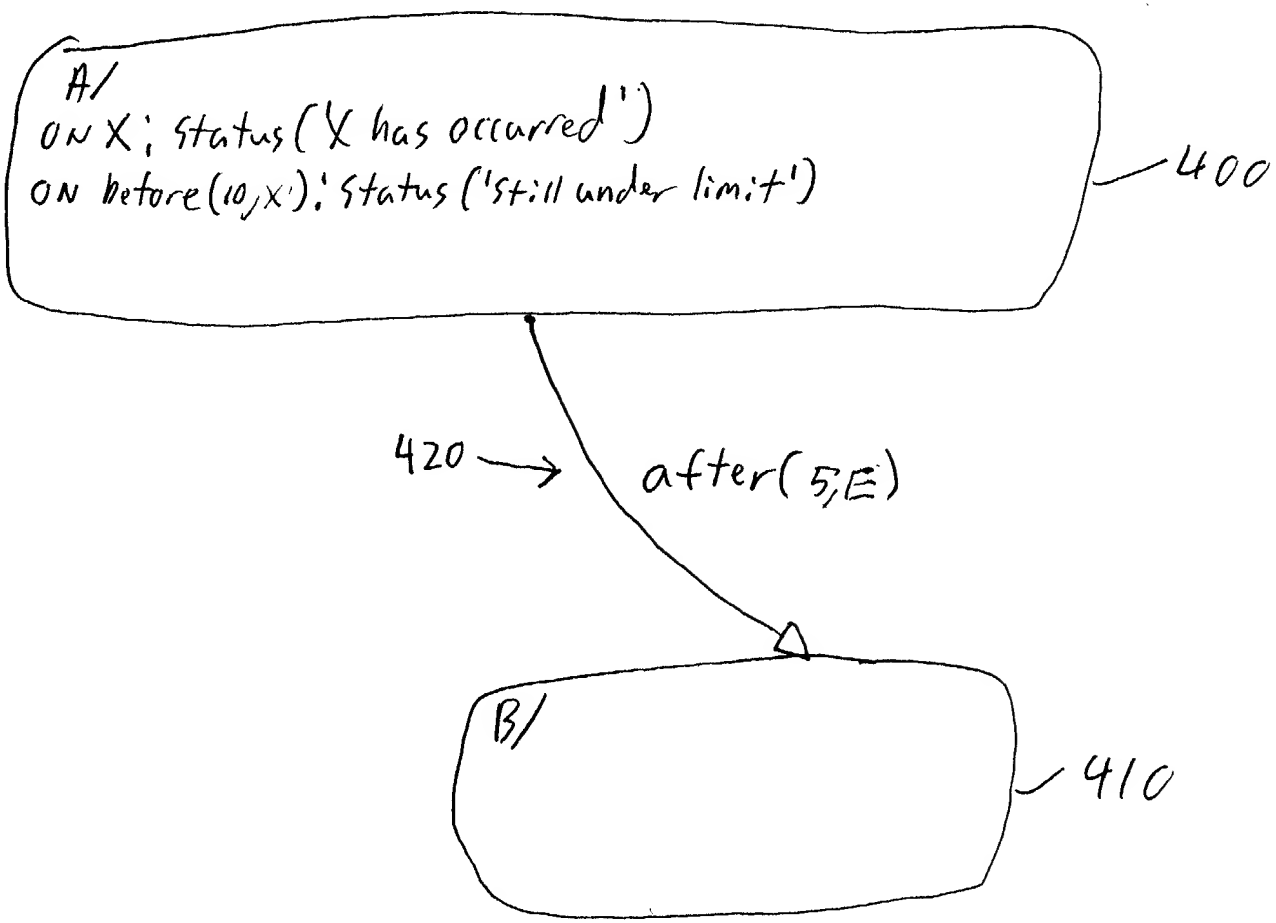
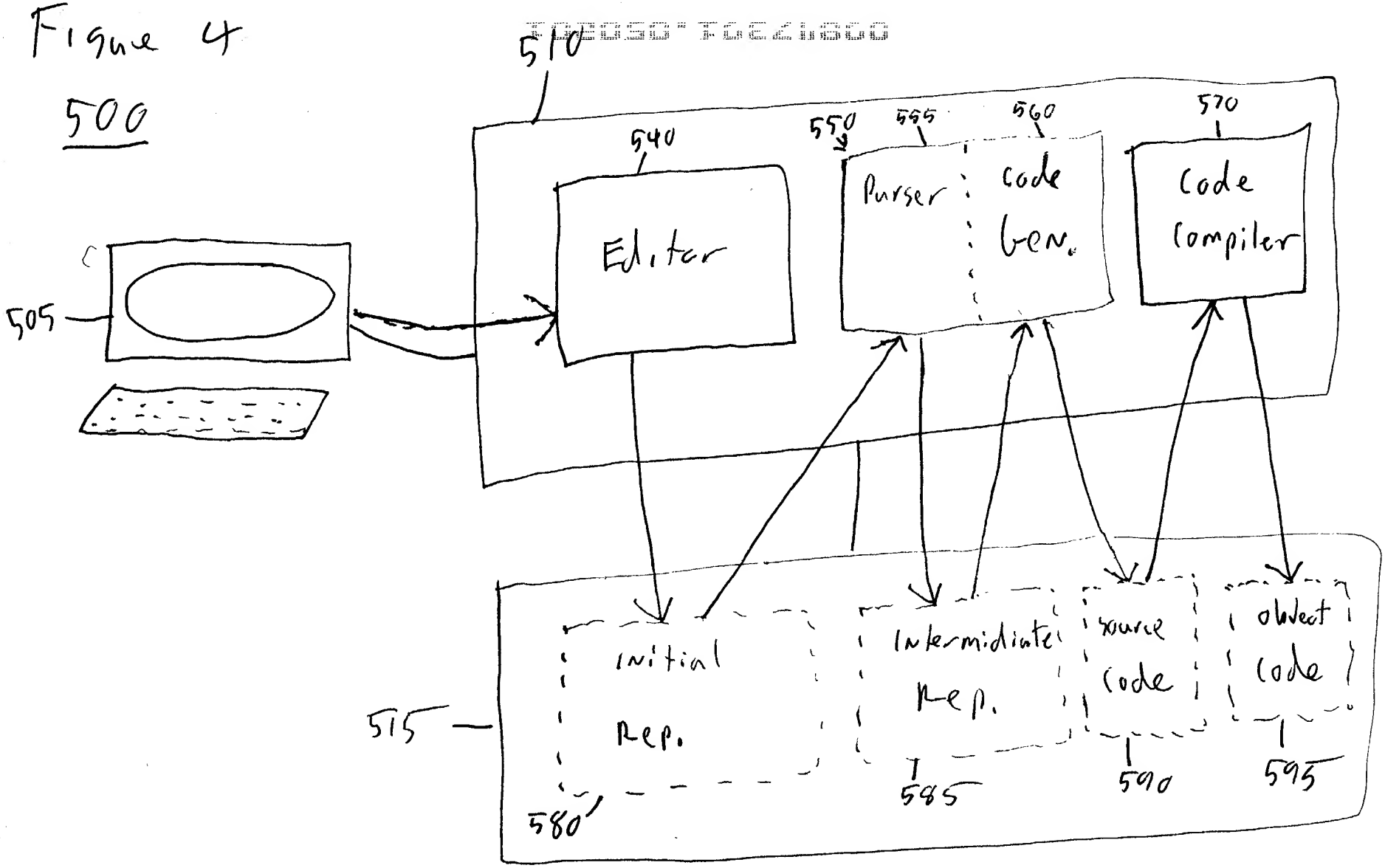

A/
ON X; status('X has occurred')
ON before(10,X): status('still under limit')
400
420
after(5,E)
B/
410

Figure 4

500


510
540
550
555
560
570
Editor
Parser
code gen.
Code Compiler
505
515
initial rep.
580
Intermidiate rep.
585
source code
590
object code
595

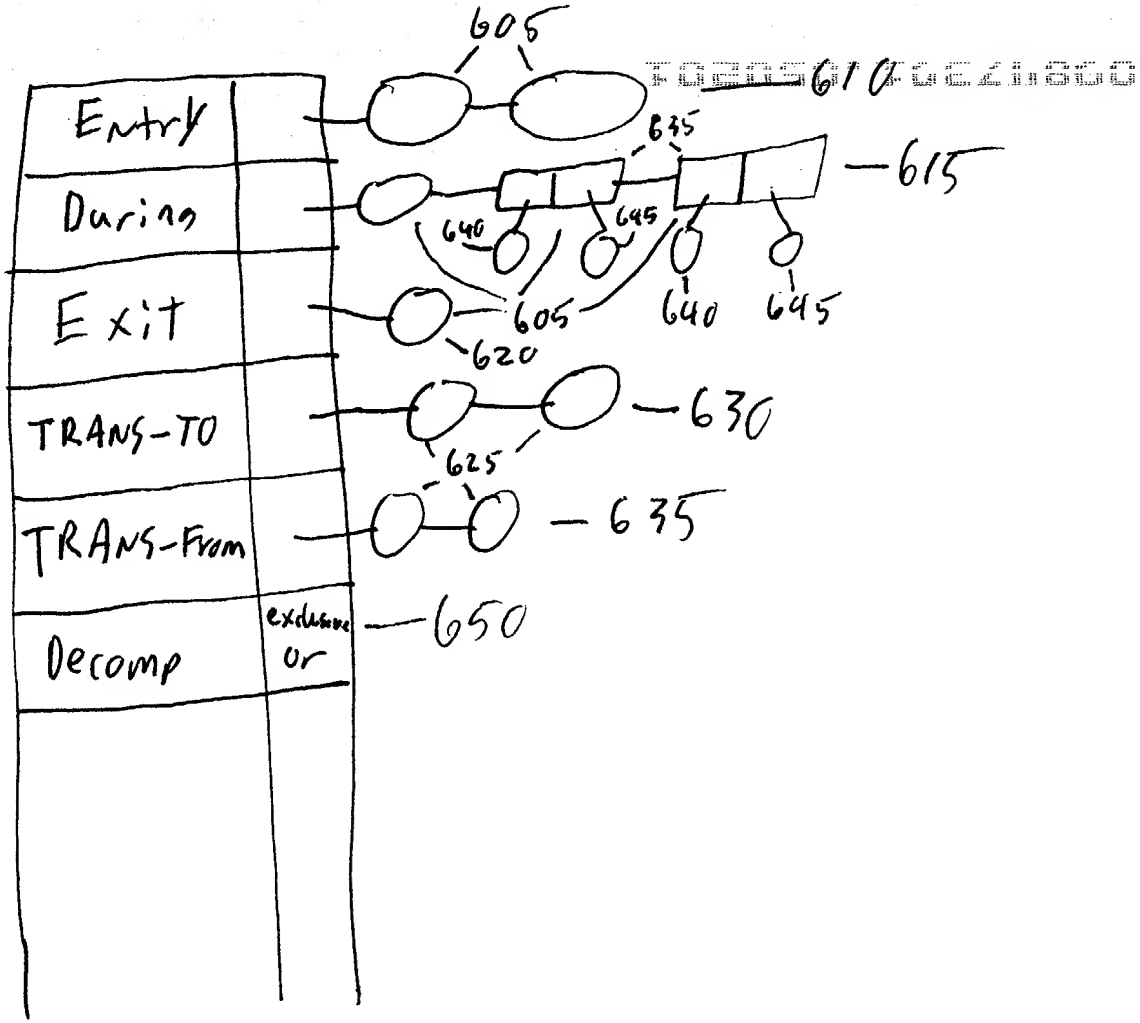

605
610
Entry
635
615
During
640
645
Exit
605
640
645
620
TRANS-TO
630
625
TRANS-From
635
exclusive or
650
Decomp

Fig. 5

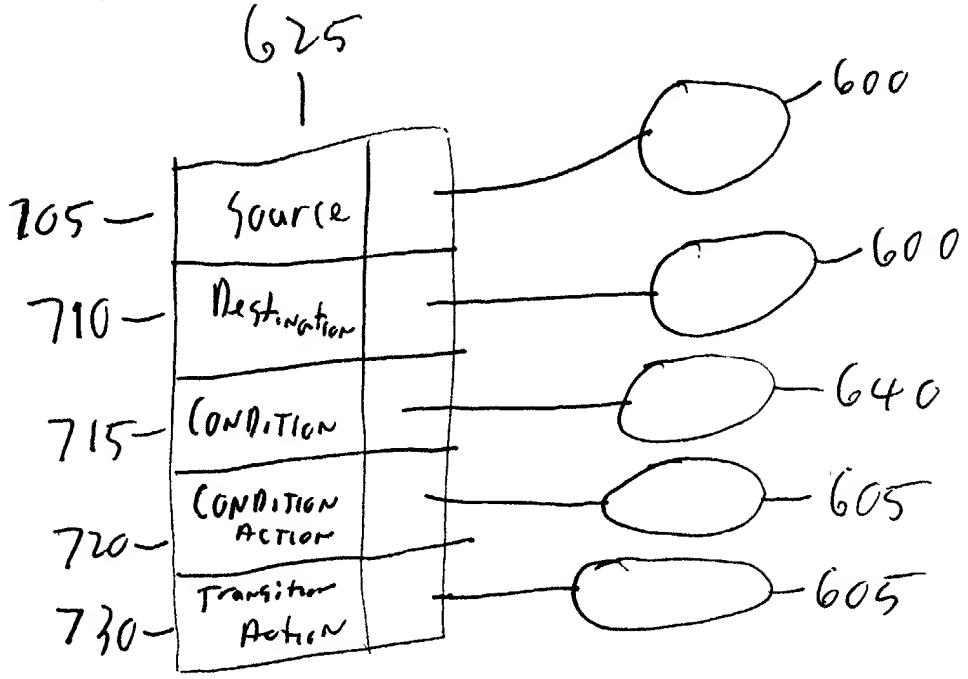

625
705
Source
710
Destination
715
CONDITION
720
CONDITION ACTION
730
Transition Action
600
600
640
605
605

Fig. 6

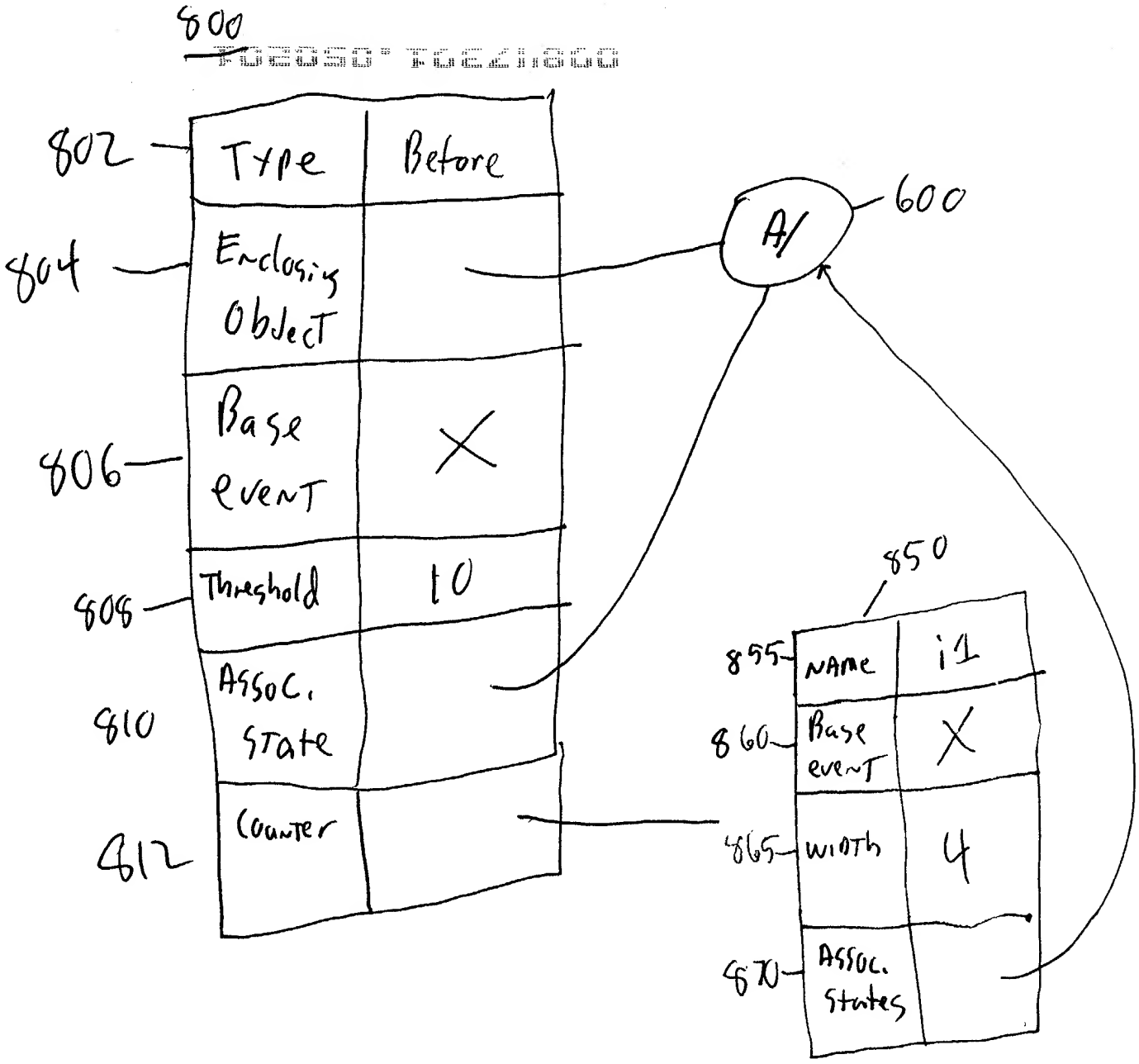

800
802
Type
Before
804
Enclosing Object
806
Base event
X
808
Threshold
10
810
Assoc. State
812
Counter
600
A1
850
855
NAME
i1
860
Base event
X
865
WIDTH
4
870
Assoc. States

Fig. 7

Fig. 8

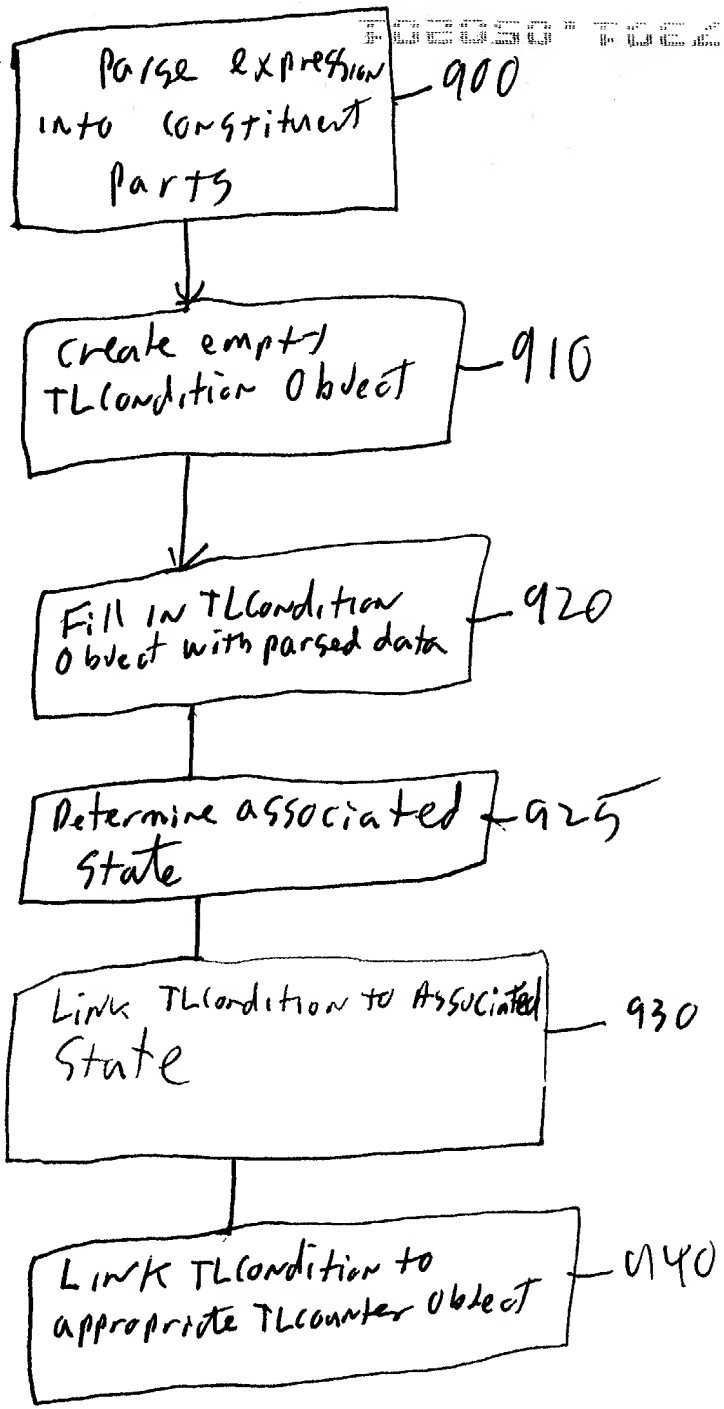

Parse expression into constituent parts
900
Create empty TLCondition Object
910
Fill in TLCondition Object with parsed data
920
Determine associated State
925
Link TLCondition to Associated State
930
Link TLCondition to appropriate TLCounter Object
940

Fig. 9

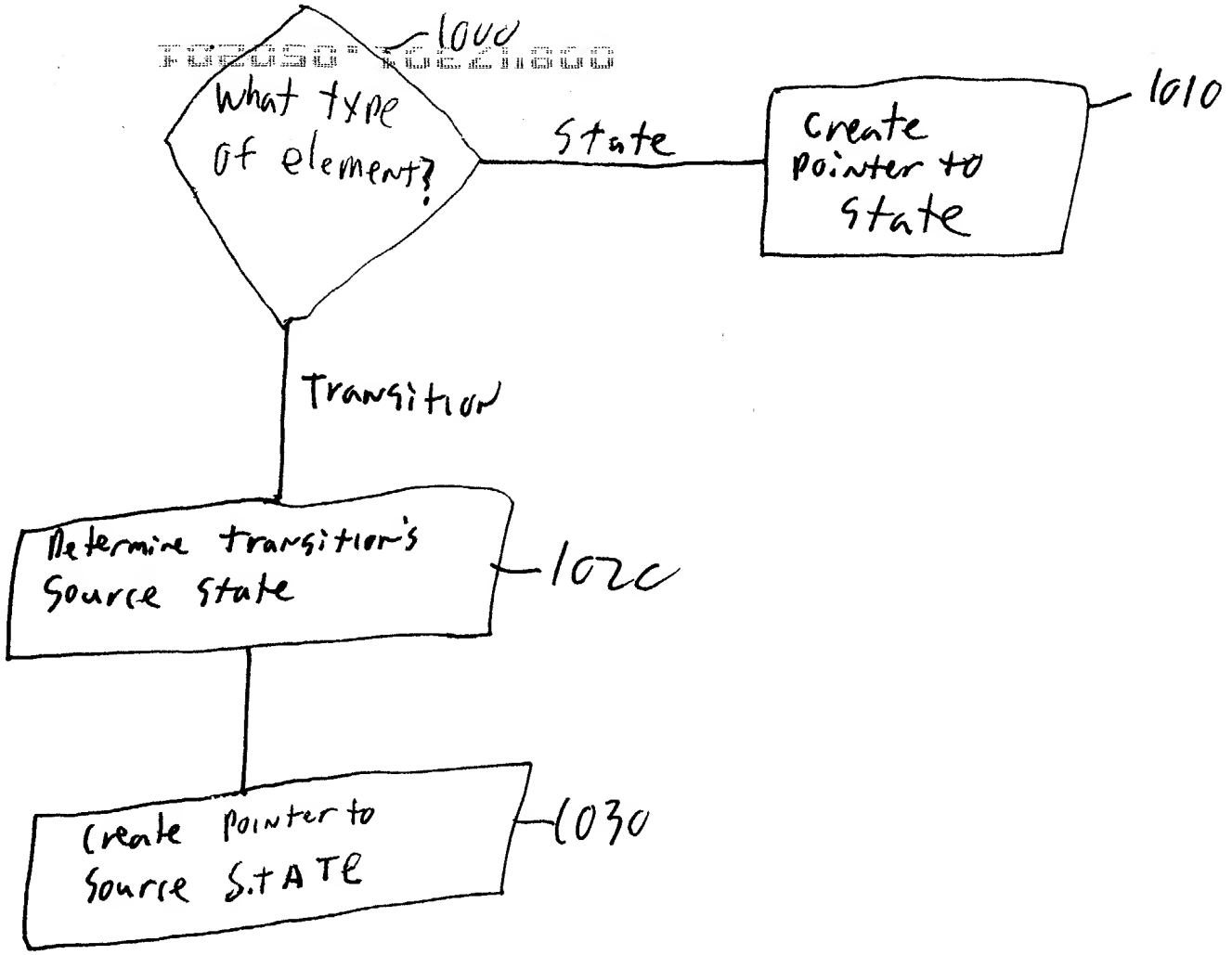

1000
What type of element?
State
Create pointer to State
1010
Transition
Determine transition's Source State
1020
Create Pointer to Source STATE
1030

Fig. 10

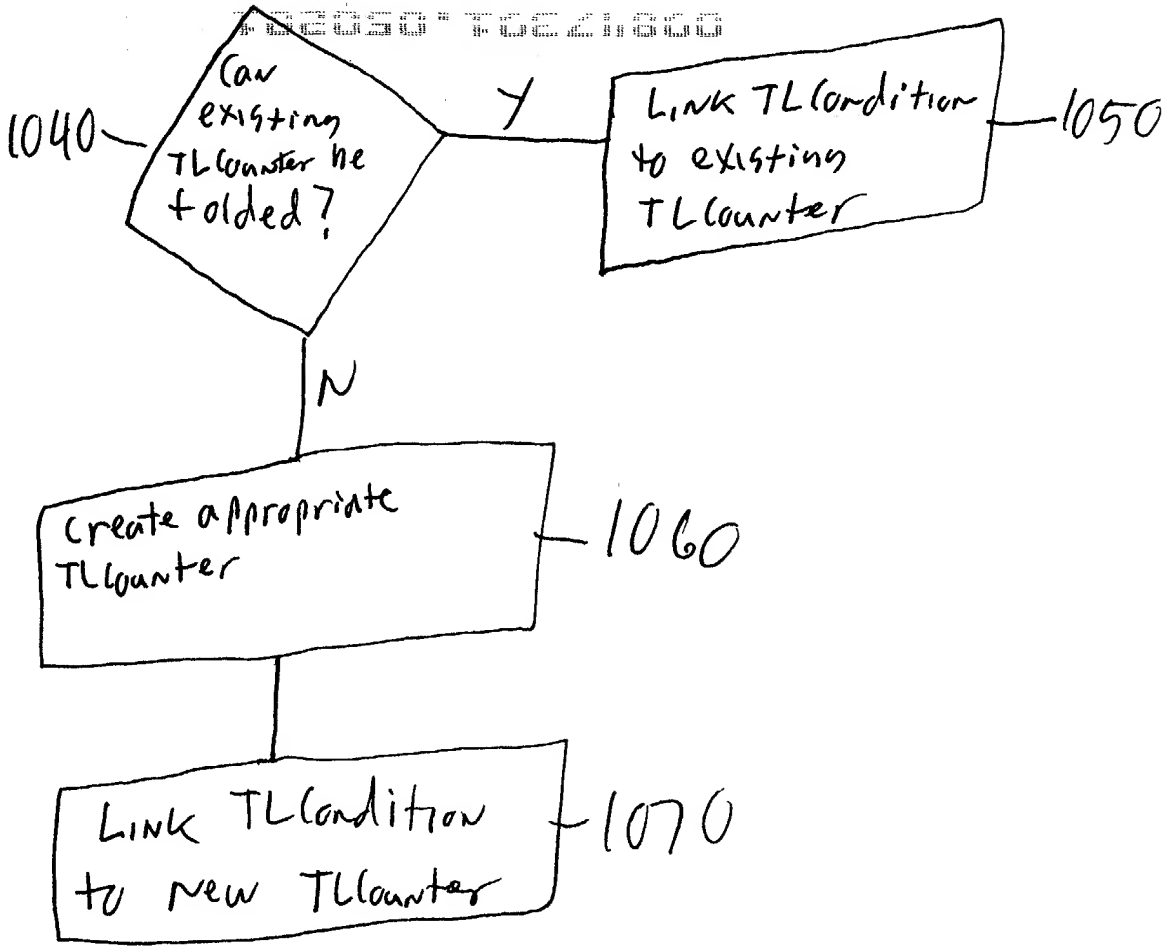

Can existing TLcounter be folded?
1040
Y
Link TLcondition to existing TLcounter
1050
N
Create appropriate TLcounter
1060
Link TLcondition to new TLcounter
1070

Figure 11

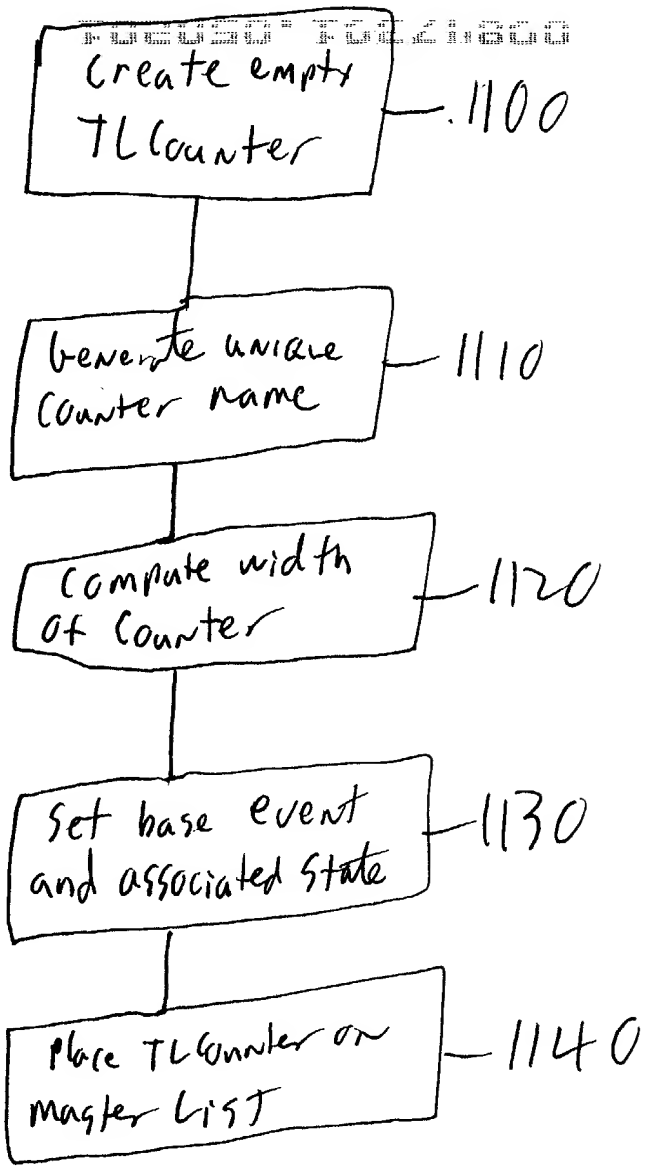

Create empty TLCounter
1100
Generate unique Counter name
1110
Compute width of Counter
1120
Set base event and associated State
1130
Place TLCounter on master List
1140

Figure 12

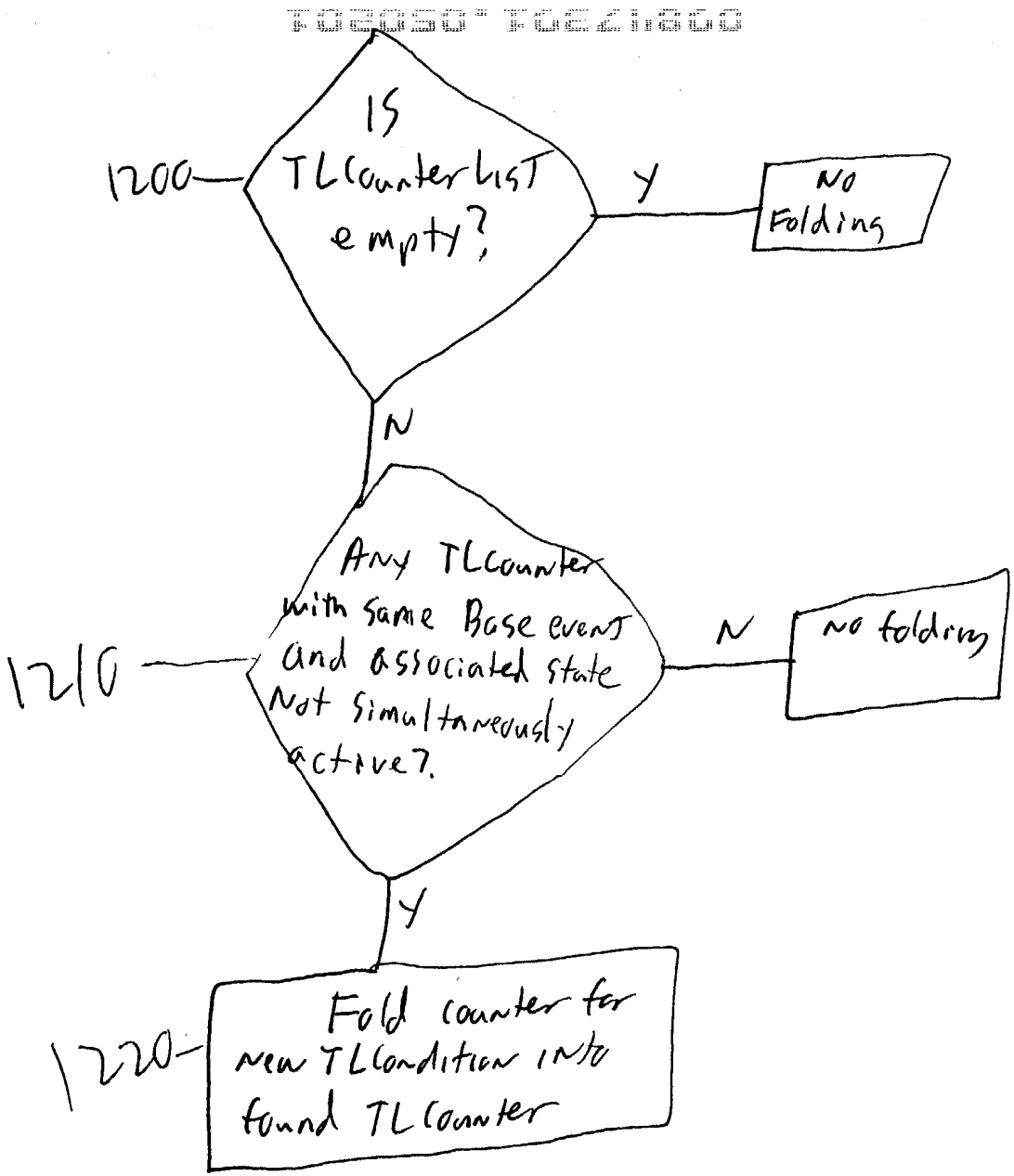

1200
Is TLCounterList empty?
Y
No Folding
N
1210
Any TLCounter with same Base event and associated state Not Simultaneously active?
N
no folding
Y
1220
Fold counter for new TLCondition into found TLCounter

Figure 13

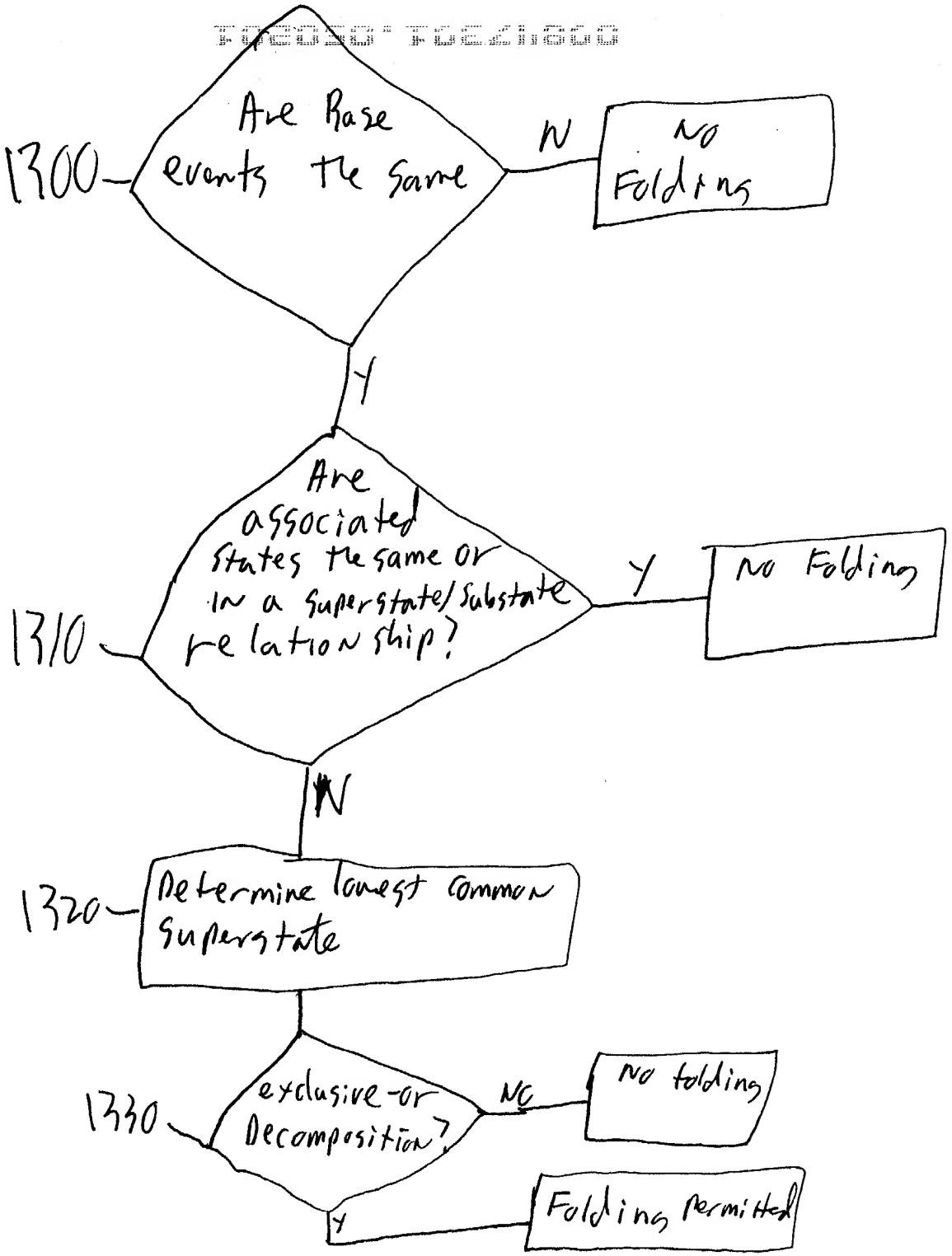

1300
Are Raise events the same
N
No Folding
Y
1310
Are associated states the same or in a superstate/substate relationship?
Y
No Folding
N
1320
Determine lowest common superstate
1330
exclusive-or Decomposition?
NO
no folding
Y
Folding permitted

Figure 14

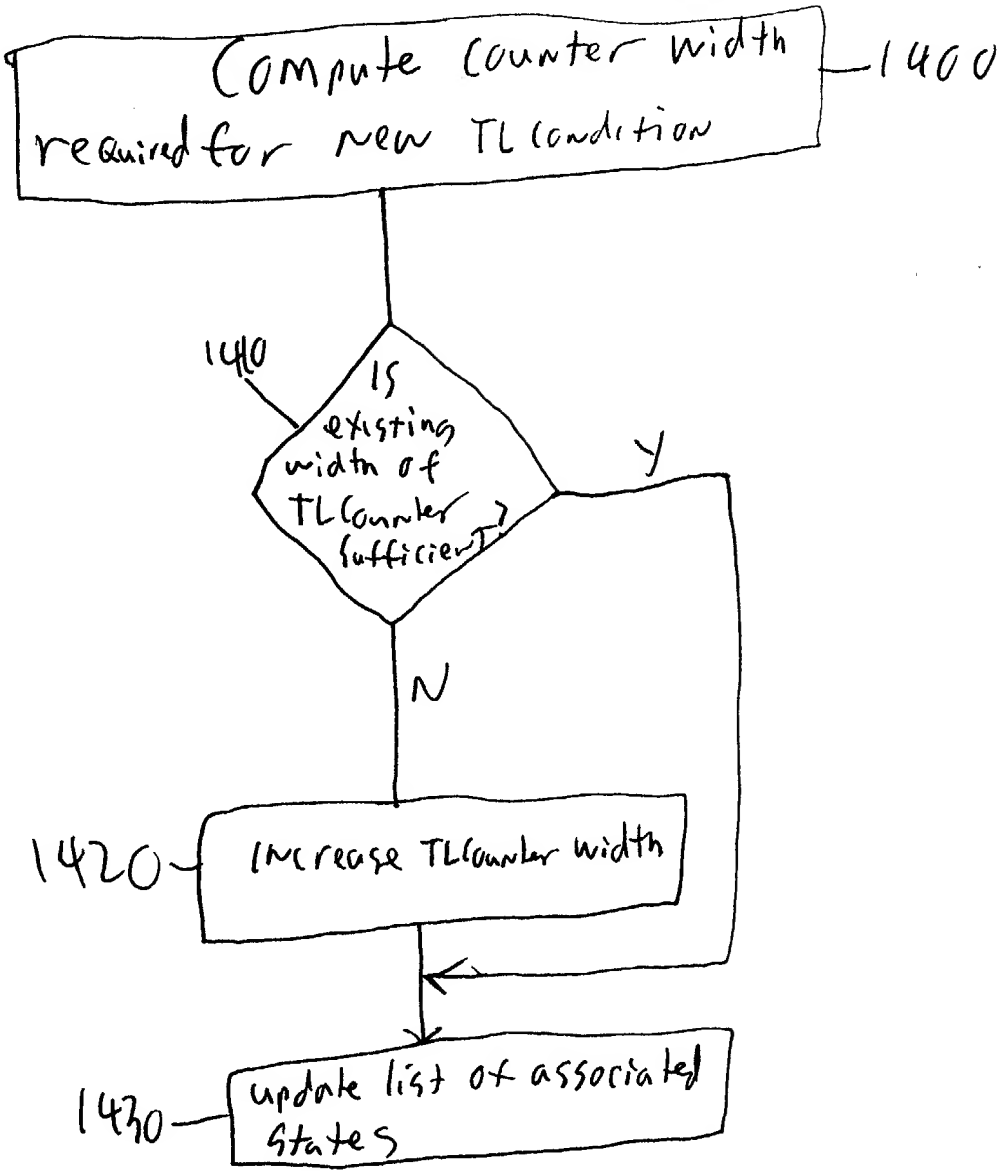

Compute counter width required for new TL condition
1400
1410
Is existing width of TL counter sufficient?
Y
N
1420
Increase TL counter width
1430
update list of associated states

Figure 15

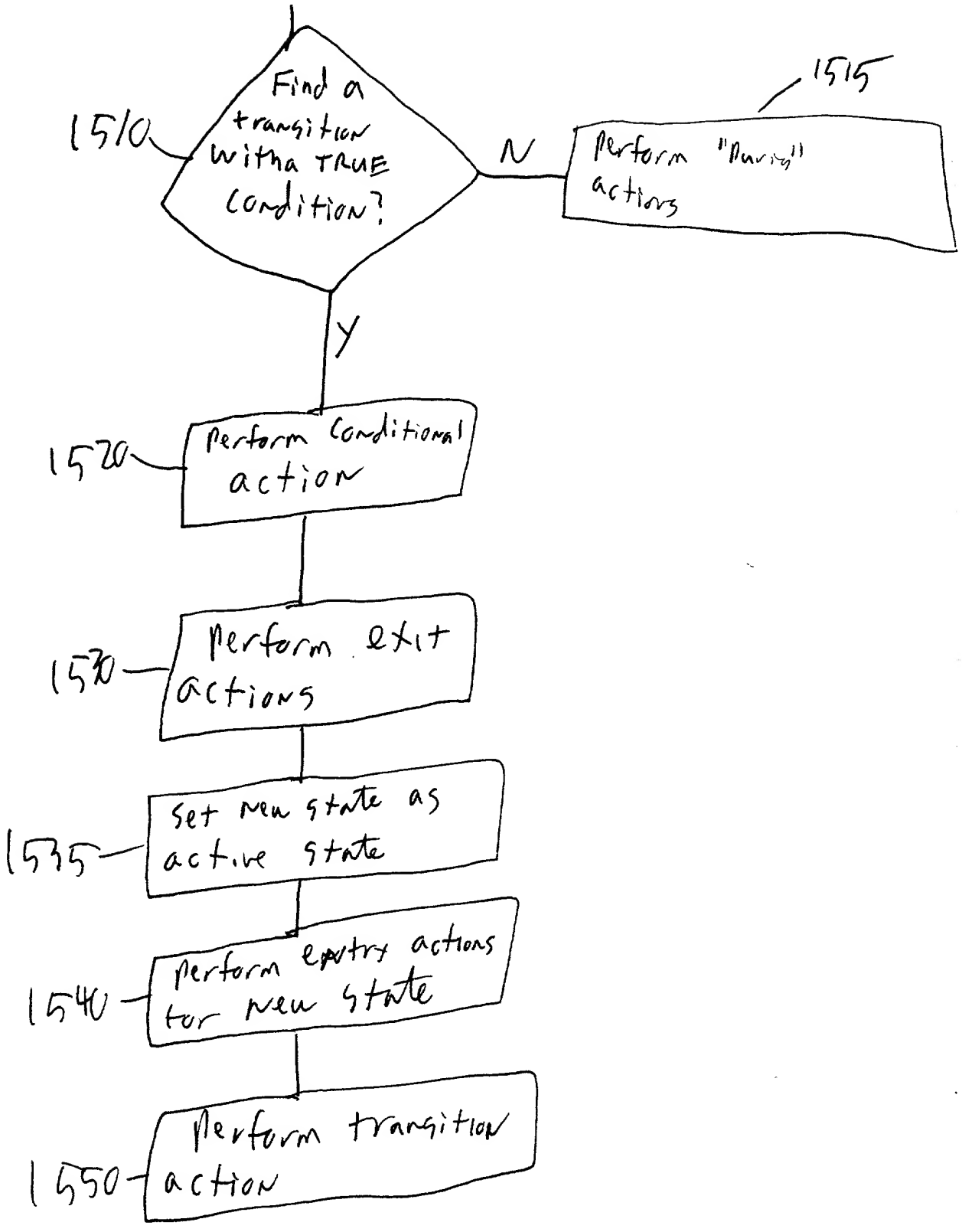
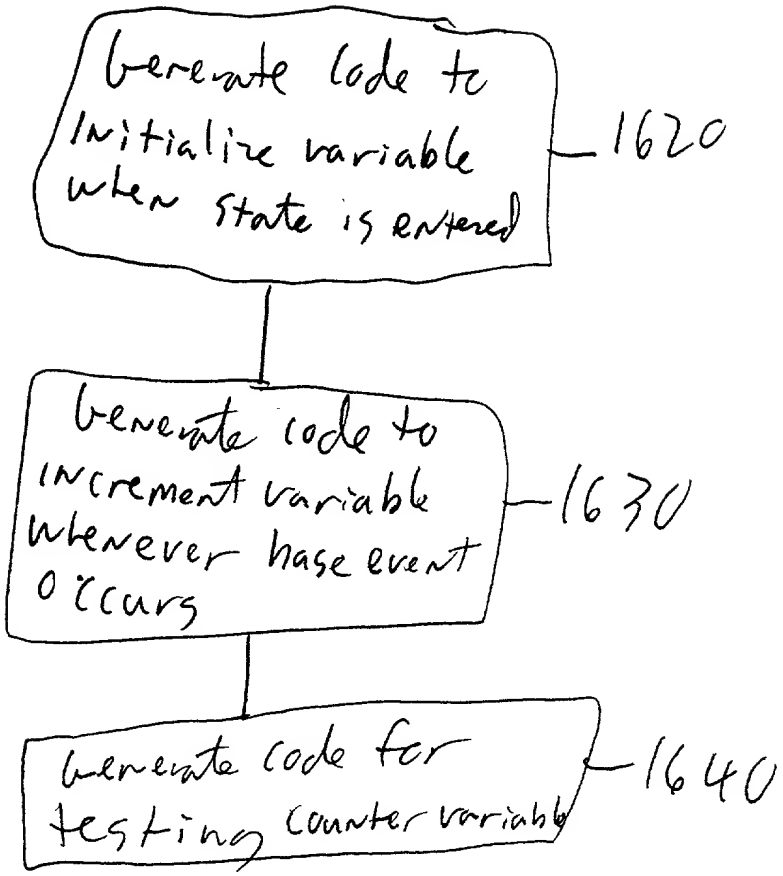

Find a transition with a TRUE condition?
1510
N
Perform "Burning" actions
1515
Y
Perform Conditional action
1520
Perform exit actions
1530
Set new state as active state
1535
Perform entry actions for new state
1540
Perform transition action
1550

Figure 16


Generate code to initialize variable when state is entered
1620
Generate code to increment variable whenever base event occurs
1630
Generate code for testing counter variable
1640

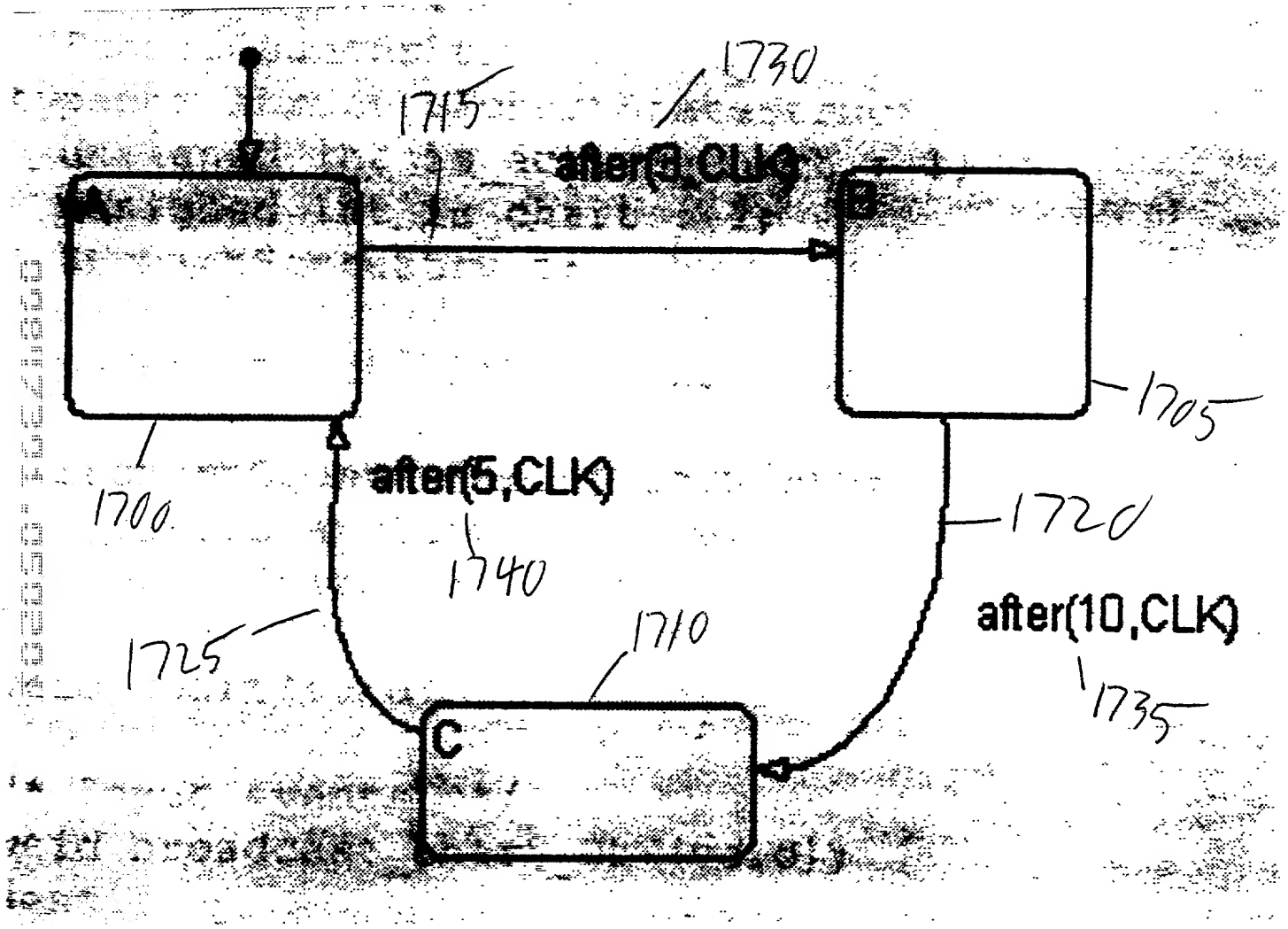

1730
1715
after(1,CLK)
A
B
1705
1700
after(5,CLK)
1720
1740
1725
1710
after(10,CLK)
1735
C

Figure 17

```
/*
 *
 * Stateflow code generation for chart:
 *    temporal_example/Chart
 *
 * Target Name                          : target
 * Model Version                        : 1.188
 * Stateflow Version                    : 4.0.3.12.00.1.000000
 * Date of code generation              : 26-Mar-2001 12:31:13
 *
 */

#ifndef __chart_h__
#define __chart_h__
typedef struct SFchartCounterStruct{
  unsigned int i1 : 4;
}SFchartCounterStruct;
typedef struct SFchartStateStruct{
  unsigned int is_active_chart : 1;
  unsigned int is_chart : 2;
} SFchartStateStruct;

typedef struct SFchart_InstanceStruct {

  SFchartCounterStruct Counters;
  SFchartStateStruct State;
} SFchartInstanceStruct;

void chart(void);

/* Input events: */
void broadcast_chart_CLK(void);
#endif
```

1800

1820

1830

1840

1810

Figure 18

```
 *    temporal_example/Chart
 *
 * Target Name                          : target
 * Model Version                        : 1.188
 * Stateflow Version                    : 4.0.3.12.00.1.000000
 * Date of code generation              : 26-Mar-2001 12:31:13
 *
 */

#include "temporal_example_target.h"
#include "chart.h"

#define IN_NO_ACTIVE_CHILD              (0)
#define IN_c1_s1_A                      1
#define IN_c1_s2_B                      2
#define IN_c1_s3_C                      3
#define event_CLK                       0
static SFchartInstanceStruct chartInstance;
void chart(void);

void chart(void)
{

  {
    /* During: Chart */
    if(_sfEvent_temporal_example_ == event_CLK) {
      if(chartInstance.Counters.i1<0xfU) {
        chartInstance.Counters.i1++;
      }
    }
    if(chartInstance.State.is_active_chart == 0) {
      /* Entry: Chart */
      chartInstance.State.is_active_chart = 1;
      /* Entry: A */
      chartInstance.State.is_chart = IN_c1_s1_A;
      chartInstance.Counters.i1=0;
    } else {
      switch(chartInstance.State.is_chart) {
       case IN_c1_s1_A:
        /* During: A */
        if((_sfEvent_temporal_example_ == event_CLK) &&
         (chartInstance.Counters.i1 >= 3)) {
          /* Exit: A */
          /* Entry: B */
          chartInstance.State.is_chart = IN_c1_s2_B;
          chartInstance.Counters.i1=0;
        }
        break;
       case IN_c1_s2_B:
        /* During: B */
        if((_sfEvent_temporal_example_ == event_CLK) &&
         (chartInstance.Counters.i1 >= 10)) {
          /* Exit: B */
          /* Entry: C */
          chartInstance.State.is_chart = IN_c1_s3_C;
          chartInstance.Counters.i1=0;
        }
```

1910

1920

1930

1940

1950

1960

Figure 19A

```
        break;
       case IN_c1_s3_C:
        /* During: C */
        if((_sfEvent_temporal_example_ == event_CLK) &&
         (chartInstance.Counters.i1 >= 5)) {
          /* Exit: C */
          /* Entry: A */
          chartInstance.State.is_chart = IN_c1_s1_A;
          chartInstance.Counters.i1=0;
        }
        break;
      }
    }
  }
}

void broadcast_chart_CLK(void)
{

  {
    int8_T previousEvent;
    previousEvent = _sfEvent_temporal_example_;
    _sfEvent_temporal_example_ = event_CLK;
    chart();
    _sfEvent_temporal_example_ = previousEvent;
  }
}
```

1970

1900

Figure 19B

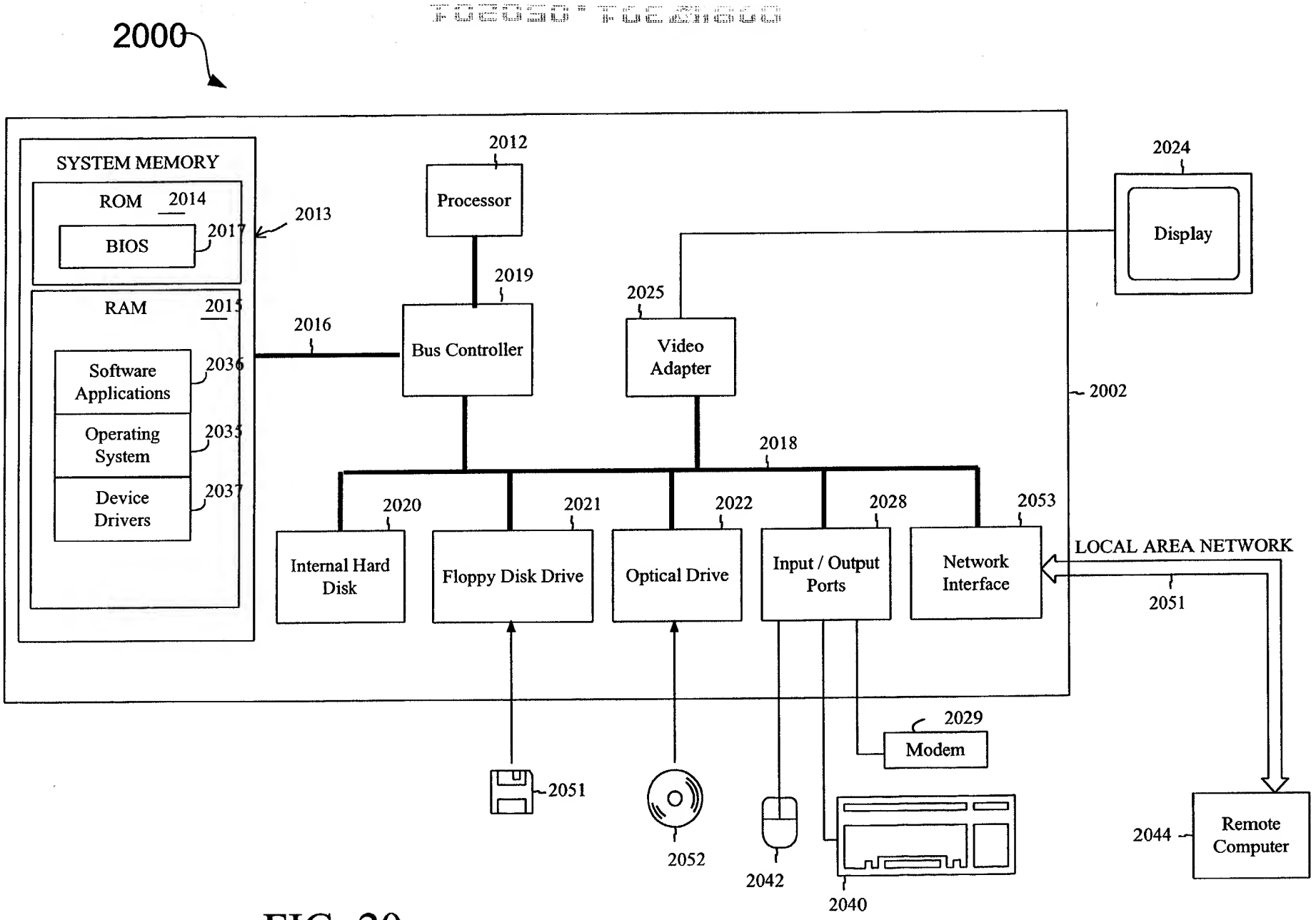

2000
SYSTEM MEMORY
ROM 2014
BIOS
2017
2013
RAM 2015
Software Applications
2036
Operating System
2035
Device Drivers
2037
2012
Processor
2019
2016
Bus Controller
2025
Video Adapter
2024
Display
2002
2018
2020
2021
2022
2028
2053
Internal Hard Disk
Floppy Disk Drive
Optical Drive
Input / Output Ports
Network Interface
LOCAL AREA NETWORK
2051
2029
Modem
2051
2052
2042
2040
2044
Remote Computer

FIG. 20